Date=28/07/2020
Lecture By=Shubham Joshi
Subject ⇒ Sorting-2

| IN PREVIOUS LECTURE **(QUICK RECAP) Date-27/07/2020** | **In Today's Lecture (Overview)** |
|---|---|
| ⇒ Backtracking In python<br>⇒ Question That Are Based on BackTracking<br>⇒ Mcq's<br>⇒ Questions For Self Practice / CC For the Day | Merge Sort In Python<br>Question related Sort Merge<br>Mcqs<br>Questions For Self Practice |

# Merge Sort In Python

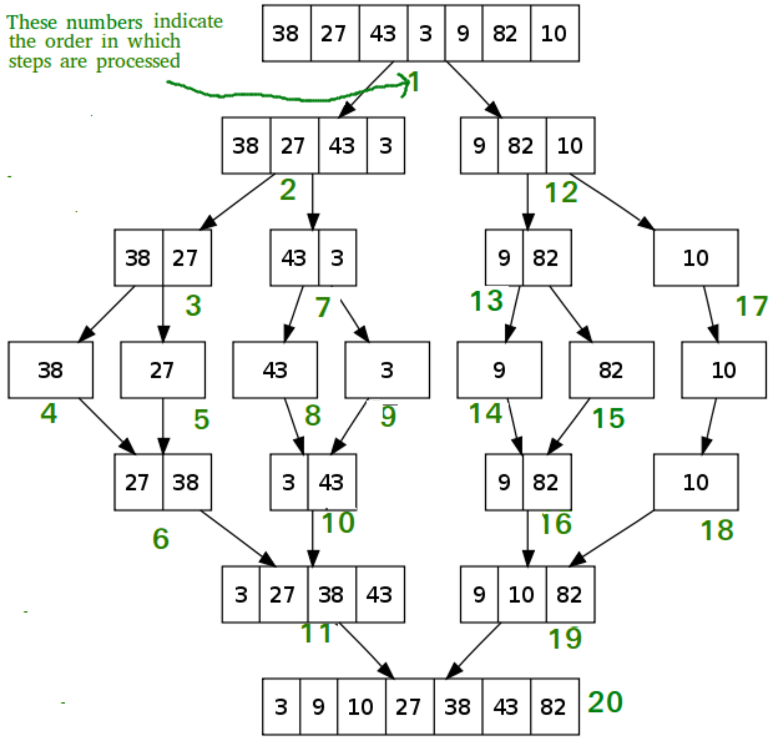
Merge Sort is a **Divide and Conquer** algorithm.

It divides the **input array in two halves**, calls itself for the two halves and then merges the two sorted halves.

Time complexity of Merge sort is **n(logn)**

“Click Here” to know more about It

For video Tutorial “Click Here”

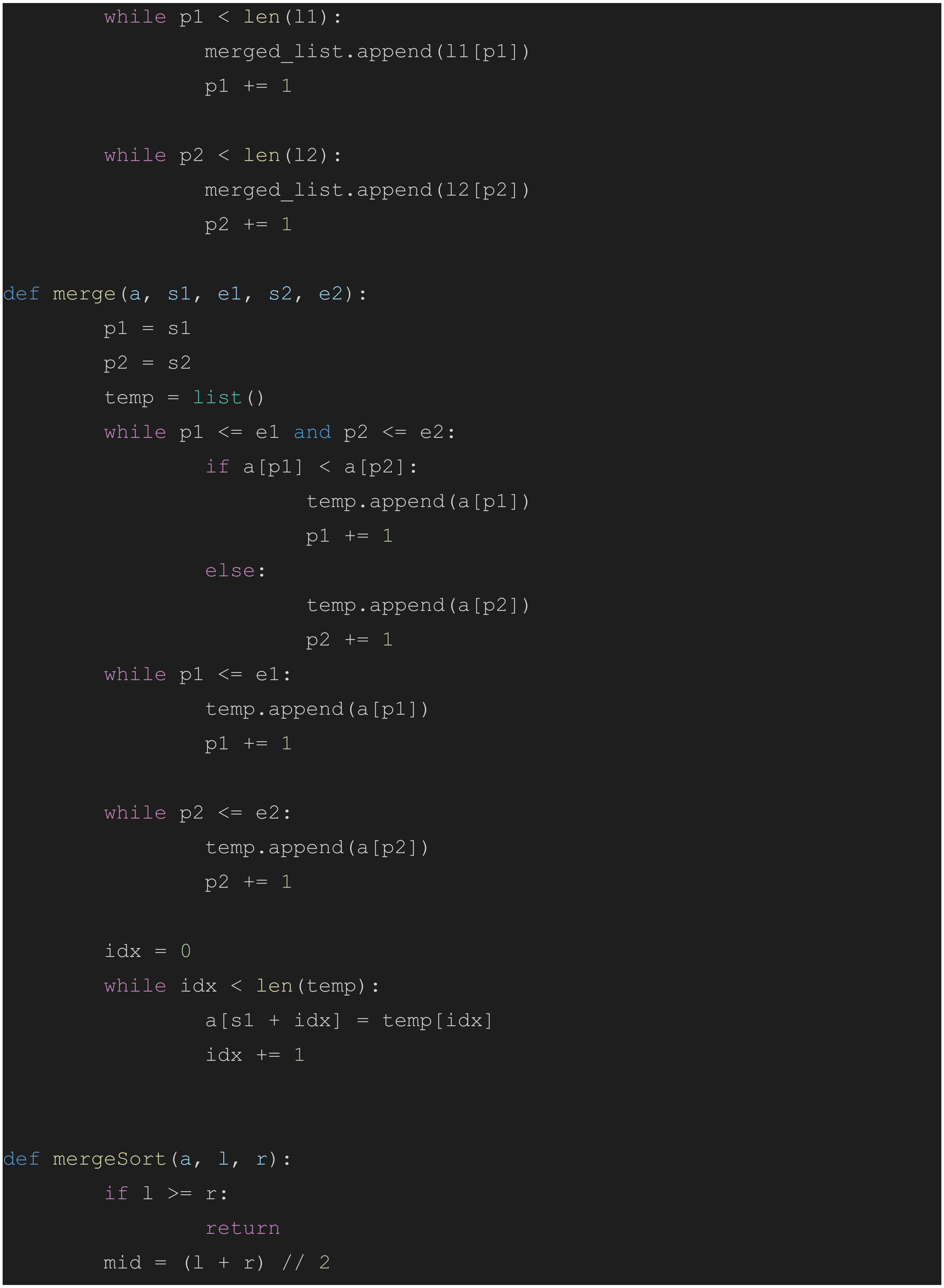


# Question related Sort Merge

### Given 2 sorted arrays You Have to merge them using Sort Merge

### Code

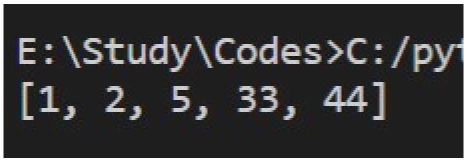
```
def merge(l1, l2):
        p1 = 0
        p2 = 0

        merged_list = list()
        while p1 < len(l1) and p2 < len(l2):
                if l1[p1] < l2[p2]:
                        merged_list.append(l1[p1])
                        p1 += 1
                else:
                        merged_list.append(l2[p2])
                        p2 += 1
```



```
    while p1 < len(l1):
        merged_list.append(l1[p1])
        p1 += 1

    while p2 < len(l2):
        merged_list.append(l2[p2])
        p2 += 1

def merge(a, s1, e1, s2, e2):
    p1 = s1
    p2 = s2
    temp = list()
    while p1 <= e1 and p2 <= e2:
        if a[p1] < a[p2]:
            temp.append(a[p1])
            p1 += 1
        else:
            temp.append(a[p2])
            p2 += 1
    while p1 <= e1:
        temp.append(a[p1])
        p1 += 1

    while p2 <= e2:
        temp.append(a[p2])
        p2 += 1

    idx = 0
    while idx < len(temp):
        a[s1 + idx] = temp[idx]
        idx += 1


def mergeSort(a, l, r):
    if l >= r:
        return
    mid = (l + r) // 2
```

```
        mergeSort(a, l, mid)
        mergeSort(a, mid + 1, r)
        merge(a, l, mid, mid + 1, r)

if __name__ == '__main__':
        l = [5,2,1,33,44]        mergeSort(l, 0, len(l) - 1)
        print(l)
```

**Output**



```
E:\Study\Codes>C:/pyt
[1, 2, 5, 33, 44]
```

### We Only Discussed This Topic’s One question as this topic is very important

# Mcqs

**1.What is the worst time complexity of merge sort ?2**

**a.O(n)**

**b.O(nlogn)**

**c.O(1)**

**2.What will be the best time complexity to find intersection of 2 arrays ?**

**a.O(N)**

**b.O(N2)**

**c.O(NLOGN)**

**3.which does merge Works11?2**

**a.divide and conquer**

**b.recursion**

**4.what does extend do in a list ?**

**a.appends another list**

**b.makes it a matrix**

**c.appends an element**

# Questions For Self Practice

**Q1. Implement merge sort on your own and analyze its time complexity.**

**Q2. https://practice.geeksforgeeks.org/problems/merge-two-sorted-arrays/0**

**Q3. https://practice.geeksforgeeks.org/problems/sort-an-array-of-0s-1s-and-2s/0**